Appendix B

HP-UX 10.x Security Checklist

Topic: **AUDIT**
SubTopic:

Objective 14
Ensure the audit subsystem is configured correctly and securely and auditing is enabled.

Rationale:
UNIX maintains a number of log files that keep track of when users log in and which commands they run. These log files form the basis of UNIX's auditing system. Auditing can be enabled or disabled. It should always be enabled for a secure system.

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
Invoke SAM by typing:

```
/usr/sbin/sam
```

Select "`Auditing and Security (Trusted System)`." The upper portion of the window specifies if auditing is on or off.

Expected Results:
Verify that auditing is turned on. If auditing is not enabled select "Actions" then "Turn Auditing On" from the pull down menus.

Note: A simple test to determine if the workstation has been converted to a trusted system is to look for the "last successful/unsuccessful" login message that is displayed at login by a trusted system.

Comments:
Auditing should be enabled.

Various audit tasks can be completed manually by using the following commands:

audsys: Starts/Halts auditing; displays audit file information
audusr: Selects users to be audited
audevent: Displays/displays audit event status
audomon: Sets audit file size parameters
audisp: Displays the audit record.

See online documentation for additional information on these commands.

Topic: **AUDIT**
SubTopic:

Objective 15
Ensure audit is configured to collect required audit events (login and logout, use of privileged commands, application and session initiation, use of print command, DAC permission modification, export to media, unauthorized access attempts to files...).

Rationale:
DII COE SRS Requirement:
3.2.2.5 At a minimum, the following audit events shall be audited:
3.2.2.5.1 Login (unsuccessful and successful) and Logout (successful)
3.2.2.5.2 Use of privileged commands (unsuccessful and successful)
3.2.2.5.3 Application and session initiation (unsuccessful and successful)
3.2.2.5.4 Use of print command (unsuccessful and successful)
3.2.2.5.5 Discretionary access control permission modification (unsuccessful and successful)
3.2.2.5.6 Export to media (successful)
3.2.2.5.7 Unauthorized access attempts to files (unsuccessful)
3.2.2.5.8 System startup and shutdown (unsuccessful and successful).

Test Actions:
Step: 1
Required Action:
Invoke SAM by typing:

```
/usr/sbin/sam
```

Expected Results:

Comments:
Note: A simple test to determine if the workstation has been converted to a trusted system is to look for the "last successful/unsuccessful" login message that is displayed at login by a trusted system.

Step: 2
Required Action:
Highlight "`Auditing and Security (Trusted System)`."

Expected Results:

Comments:

Step: 3
Required Action:
Highlight "`Modify (or view) Auditing Options -Events.`"

Expected Results:
At the minimum the following events should be selected:

login - Logs all logins and logouts.
moddac - Logs all modifications of object's Discretionary Access Controls.
modaccess - Logs all access modifications other than DAC.
admin - Logs all administrative and privileged events.

Comments:

Topic: **AUDIT**
SubTopic:

Objective 17
Verify the system provides the SGSO with a capability to select and enable auditable events including use of I&A, introduction of objects into a user's address space, deletion of objects, trusted user actions, print use, etc.

Rationale:

DII COE SRS Requirement:
3.2.2.2 The COE shall provide the SGSO with a capability to select and enable auditable events.

3.2.2.3 The COE shall be able to audit the following types of events:

3.2.2.3.1 Use of I&A mechanisms
3.2.2.3.2 Introduction of objects into a user's address space (e.g., file open, program initiation)
3.2.2.3.3 Deletion of objects
3.2.2.3.4 Actions taken by trusted users
3.2.2.3.5 Production of printed output
3.2.2.3.6 Other security relevant events.

Test Actions:
Step: 1
Required Action:
Invoke SAM by typing:

```
/usr/sbin/sam
```

Select "Auditing and Security (Trusted System)." Select "Modify (or view) Auditing Options -System Calls." Determine what audit options are selected.

Expected Results:
The following event types are supplied by the system as default event types and become activated when the Perform Task key is pressed.

Login
Moddac
Admin

All system calls associated with a specific event type are audited automatically when that event type is selected for auditing (Hewlett Packard, 1991).

Minimally, the following set of audit events should be selected: login, moddac, modaccess, and admin.

Comments:

Topic: **AUDIT**
SubTopic:

Objective 18
Identify any users for whom auditing has been disabled.

Rationale:
An audit flag is on for all existing users at initial conversion to a trusted system. Auditing for individual users can be disabled.

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
Invoke SAM by typing:

```
/usr/bin/sam
```

Expected Results:

Comments:

Step: 2

Required Action:
Highlight "`Auditing and Security (Trusted System).`"

Expected Results:

Comments:

Step: 3

Required Action:
Select "`Modify (or View) Auditing Options-Users.`" Determine if the "Audit all users" parameter is enabled. If not, identify those accounts that do NOT have an "x" placed next to them to indicate audit enabled.

Expected Results:
The "audit all users" parameter should be selected.

Comments:
All users should be audited.

Topic: **AUDIT**
SubTopic:

Objective 19
Verify required parameters are identified for each recorded audit event including date and time of event, userid, type of event, success or failure of event, for I&A events, the origin of the request, etc.

Rationale:

DII COE SRS Requirement:
3.2.2.4 For each recorded event, at a minimum the audit record shall identify:

3.2.2.4.1 Date and time of the event
3.2.2.4.2 UserID
3.2.2.4.3 Type of event
3.2.2.4.4 Success or failure of the event
3.2.2.4.5 For I&A events, the origin of the request (e.g., terminal ID)
3.2.2.4.6 For events that introduce an object into a user's address space, and for object deletion events, the
name of the object, and in MLS systems, the object's security level.

Test Actions:
Step: 1

Required Action:
The native HP audit subsystem stores the audit trails in the "/.secure" directory. Use the "audisp" command to browse one of the audit data files.

Expected Results:
The required parameters listed in the objective above should be included as part of the audit logs.

Comments:

Topic: **AUDIT**
SubTopic: **Protection of Audit Data**

Objective 25
Verify the audit data is protected by the system so that access to it is limited to only those authorized to view the audit data.

Rationale:

DII COE SRS Requirement:
3.2.2.1.1 The audit data shall be protected by the system so that access to it is limited to those who are authorized to view audit data.

Test Actions:
Step: 1

Required Action:
By default, HP UX stores its log files in /.secure/etc/audfile1 and /.secure/etc/audfile1. Check the /etc/rc.config.d/auditing file which contains the audit parameters. Determine the filename of each audit file. For each "filename", as a NON-privileged user, type the following commands:

```
ls -ldb "filename"
more "filename"
```

Expected Results:
Each command should cause an error message to be returned.

Comments:

Topic: **AUDIT**
SubTopic:

Objective 23
Verify the system provides an auditing function capable of accepting application level audit logging requests and a standard audit format is provided for use in application level auditing.

Rationale:

DII COE SRS Requirement:
3.2.2.8 The COE shall provide an auditing function capable of accepting application level audit logging requests.
3.2.2.8.1 The COE shall provide a standard audit format (e.g., syslog format) for use in application level auditing.

Test Actions:
Step: 1
Required Action:
Type the following command:

```
ps -eaf | grep syslog
```

Expected Results:
Output on the screen should resemble the following:

```
$ps -eaf | grep syslog
    root   161     1 53   Jul 29 ?        0:01 /usr/sbin/syslogd
   cisso   893   427  9 14:07:06 pts/2    0:00 grep syslog
$
```

Comments:
Check to ensure that this works on HP.
Taken from Solaris 2.4, should work.

Topic: **AUDIT**
SubTopic: **Protection of Audit Data**

Objective 26
Verify the audit data is protected from change or deletion by general users.

Rationale:

DII COE SRS Requirement:
3.2.2.1.2 The audit function shall be protected from change or deletion by general users.

Test Actions:
Step: 1

Required Action:
By default, HP UX stores its log files in /.secure/etc/audfile1 and /.secure/etc/audfile1. Check the /etc/rc.config.d/auditing file which contains the audit parameters. Determine the name of each audit file. For each "filename", as a NON-privileged user, type the following commands:

```
vi "filename"
```

Expected Results:
Each command should cause an error message to be returned.

Comments:

Topic: **Availability**
SubTopic:

Objective 51
Verify the system provides the capability to perform system and database backups on a periodic basis.

Rationale:

DII COE SRS Requirement:
3.2.3.4 The COE shall provide the capability to perform system and database backups on a periodic basis.

Test Actions:
Step: 1

Required Action:
View the crontab file to determine whether the system is backed up automatically on a scheduled basis. From the system logbook or the System Administrator determine when the last system backup was performed and if backups are regularly performed. Determine if the backup tapes were labeled correctly.

Expected Results:
Backups are regularly performed either by cron jobs or by operational procedures.

Comments:

Topic: **CRON JOBS**
SubTopic: **Permissions**

Objective 129
Verify cron has been securely configured. Determine which form of cron is used on the system (see rationale for cron forms).

Rationale:
UNIX has programs and systems that run automatically. Many of these systems require special privileges. If an attacker can compromise these systems, he may be able to gain direct unauthorized access to other parts of the operating system, or plan a back door to gain access at a later time.

There are three forms of crontab files. The oldest form has a line with a command to be executed whenever the time field is matched by the cron daemon. To execute the commands from this old-style crontab file as a user other than root, it is necessary to make the command listed in the crontab file use the su command.

The second form of the cron file has an extra field that indicates on whose behalf the command is being run.

The third form of cron protects directories with a separate crontab file for each user. The cron daemon examines all the files and dispatches jobs based on the user owning the file.

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Browse the crontab file. Observe the format and determine which form of cron is used on the system. Perform an `ls -ldgb` on each file referenced in the crontab file to verify that none of the files are world writeable.

Expected Results:
All directories are not world or group writeable.

Comments:

Step: 2
Required Action:
Type in the following commands:

```
#/bin/find /usr/spool/cron/crontabs -type f -exec ls -ldb {} \; \
     -exec /usr/bin/more {} \;
#/bin/find /usr/spool/cron/atjobs -type f -exec ls -ldb {} \; \
     -exec /usr/bin/more {} \;
```

Topic: **CRON JOBS**
SubTopic: **Permissions**

Expected Results:
All user crontab files are owned by the correct user and group, all files that are referenced in a users crontab file, or that are referenced by files in the crontab file are not world or group writeable, and the cron job tasks are appropriate.

Comments:
Ensure root cron job files do NOT source any other files not owned by root or which are group or world writeable**.**This is done by TIGER, and maybe COPS and SPI.

Step: 3

Required Action:

Perform an `ls -ldg` and more on each file referenced in each crontab file to verify that none of the files are world writeable (check directories in the path of the referenced files also).

Expected Results:
All files that are referenced in the crontab file, or that are referenced by files in the crontab file are not world or group writeable and contain valid entries.

Comments:

Step: 4

Required Action:
Type in the following commands:

```
#ls -ldb /usr/lib/cron/log
#/usr/ucb/more /var/cron/log
```

Expected Results:
The cron log is not world or group writeable, and the cron jobs logged have been approved.

Comments:

Step: 5

Required Action:
Type in the following commands:

```
#ls -ldb /var/adm/cron/cron.allow
#/usr/ucb/more /var/adm/cron/cron.allow
#ls -ldb /var/adm/cron/cron.deny
#/usr/ucb/more /var/adm/cron/cron.deny
```

Expected Results:
The cron.allow and cron.deny files are owned by root, are NOT world writeable, and contain the correct entries.

Comments:

Topic: DAC
SubTopic:

Objective 59
Verify the lock out function is available for users to manually lock their terminals and users are required to re-authenticate themselves to unlock a locked terminal.

Rationale:

DII COE SRS Requirement:
3.2.4.12.4 The lock out function shall be available for users to manually invoke.
3.2.4.12.5 Users shall be required to re-authenticate themselves to unlock a locked terminal.

Test Actions:
Step: 1

Required Action:
Type in the following command:

#xlock

OR for DII COE, click on the padlock symbol on the status bar at the bottom of the screen.

Expected Results:
Screensaver appears.

Comments:
If the command results in a "xlock: not found" error, check for the presence of xlock on the system using the following command:

```
#find / -name "*xlock* -print
```

Step: 2

Required Action:
Press the Enter key and enter the Password.

Expected Results:
The password entry prompt appears and the screen unlocks.

Comments:

Topic: DAC
SubTopic:

Objective 63
Verify the system is capable of restricting access to objects based on the user's identity and on access modes (e.g., read, write, execute).

Rationale:
DII COE SRS Requirement:
3.2.4.2 The COE shall restrict access to objects based on the user's identity and on access modes (e.g., read, write, execute).

Test Actions:
Step: 1
Required Action:
As unprivileged user1, execute the following commands:

```
user1>echo ls -CFA > /tmp/file1
user1>chmod 700 /tmp/file1
user1>ls -ld /tmp/file1
user1>/usr/ucb/more /tmp/file1
```

Expected Results:
Output will look similar to the following:

```
user1>ls -ld /tmp/file1
-rwx------  1 user1             8 Oct 17 16:49 /tmp/file1*
user1>/usr/ucb/more /tmp/file1
ls -CFA
```

Comments:
Step: 2
Required Action:
As unprivileged user2 (a member of the same group), execute the following commands:

```
user2>ls  /tmp/file1
user2>/usr/ucb/more /tmp/file1
user2>echo date > /tmp/file1

user2>/tmp/file1
```

Expected Results:
Output similar to the following will be produced:

```
user2>ls -ld /tmp/file1
-rwx------  1 user1             8 Oct 17 16:49 /tmp/file1*
user2>more /tmp/file1
/tmp/file1: Permission denied
user2>echo date > /tmp/file1
/tmp/file1: Permission denied
```

```
user2>/tmp/file1
/tmp/file1: Permission denied
user2>
```

Topic: **DAC**
SubTopic:

Comments:

Step: 3

Required Action:
As unprivileged user1, execute the following commands:

```
user1>chmod 750
user1>ls  /tmp/file1
user1>/usr/ucb/more /tmp/file1
```

Expected Results:
Output will look similar to the following:

```
user1>ls -ld /tmp/file1
-rwxr-x---  1 user1             8 Oct 17 16:49 /tmp/file1*
user1>/usr/ucb/more /tmp/file1
ls -CFA
```

Comments:

Step: 4

Required Action:
As unprivileged user2 (a member of the same group), execute the following commands:

```
user2>ls  /tmp/file1
user2>/usr/ucb/more /tmp/file1
user2>echo date > /tmp/file1
user2>/tmp/file1
```

Expected Results:
Output will look similar to the following:

```
user2>ls -ld /tmp/file1
-rwxrwx---  1 user1             8 Oct 17 16:49 /tmp/file1*
user2>/usr/ucb/more /tmp/file1
ls -CFA
user2>echo date > /tmp/file1
/tmp/file1: Permission denied
user2>/tmp/file1
file1         file2          file3         file4          file5
file6         file7          file8         file9          file10
```

Comments:

Step: 5

Required Action:
As unprivileged user1, execute the following commands:

```
user1>chmod 770
user1>ls  /tmp/file1
user1>/usr/ucb/more /tmp/file1
```

Topic: DAC
SubTopic:

Expected Results:
Output will look similar to the following:

```
user1>ls -ld /tmp/file1
-rwxrwx---  1 user1             8 Oct 17 16:49 /tmp/file1*
user1>/usr/ucb/more /tmp/file1
ls -CFA
```

Comments:

Step: 6

Required Action:
As unprivileged user2 (a member of the same group), execute the following commands:

```
user2>ls  -ld /tmp/file1
user2>/usr/ucb/more /tmp/file1
user2>echo date > /tmp/file1
user2>/usr/ucb/more /tmp/file1
user2>/tmp/file1
```

Expected Results:
Output will look similar to the following:

```
user2>ls -ld /tmp/file1
-rwxrwx---  1 user1             8 Oct 17 16:49 /tmp/file1*
user2>/usr/ucb/more /tmp/file1
ls -CFA
user2>echo date > /tmp/file1
user2>/usr/ucb/more /tmp/file1
date
user2>/tmp/file1
Thu Oct 17 17:37:06 EDT 1996
```

Comments:

Step: 7

Required Action:
As unprivileged user1, execute the following commands:

```
user1>rm /tmp/file1
user1>ls /tmp/file1
```

Expected Results:
Output will look similar to the following:

```
user1>rm /tmp/file1
user1>ls -ld /tmp/file1
/tmp/file1: No such file or directory
```

Comments:

Topic: **FILE SYS SEC**
SubTopic:

Objective 81

Verify the shell used on the system resets the Internal Field Separator (IFS) variable when invoked.

Rationale:

The Internal Field Separator (IFS) variable can be set to indicate what characters separate input words. Most modern versions of the shell will reset their IFS value to a normal set of characters when invoked. Thus, shell files will behave properly. However, not all do (Garfinkel and Spafford, 1992).

Bourne shell inherits the value of its internal field separator from its environment. This can be used to obtain root access. In the Bourne shell, the IFS is the ASCII character used as a separator on the command line between command names and arguments. Normally the IFS is set to space or tab, but it can also be set by the user from environment variables. In UNIX, environmental variables are passed to child processes. The C library call popen(3) uses the Bourne shell and inherits the environment variables, including IFS. Because of this, the path passed to popen(3) can be altered so that an alternate program is executed. This means a setuid root program which uses popen(3) can be forced to run a program other than what it is intended to run.

If a root program does "popen("/bin/mail" ...)", and the IFS is set to " / ", then it runs the program "bin" with the command argument of "mail" and a userid of root. "/usr/lib/ex3.7preserve" is one of many programs you can use to exploit this. When "vi(1)" receives a hangup signal or when the command "p-reserve" is used, it executes the program "/usr/lib/ex3.7preserve", which preserves the current file you were editing and sends mail to you notifying you that your file was saved. To make certain it has permission to do this, "ex3.7 preserve" runs setuid to root. The security problem arises because when ex3.7 preserve tries to send mail to the user, it uses popen(3) to run "/bin/mail".

DII COE SRS Requirement:

Test Actions:

Step: 1

Required Action:

As an unprivileged user, insert the following text into a file named "ifs_test":

```
#!/bin/sh
# A test of the shell
cd /tmp
cat > tmp <<  E-O-F
echo "Security Vulnerability. Your shell does NOT reset the IFS
variable!"
E-O-F

cat > foo << E-O-F
echo "Your shell appears well behaved."
E-O-F

cat > test$$ <<E-O-F
/tmp/foo
```

E-0-F

Topic: **FILE SYS SEC**
SubTopic:

```
chmod 700 tmp foo test$$

PATH=.:$PATH
IFS=/
export PATH IFS

test$$

rm -f tmp foo test$$
THEN execute the following commands:
chmod 700 ifs_test
ifs_test
```

Expected Results:
Script file exists.

Comments:
SUID and SGID scripts should NEVER be used.

Step: ***2***

Required Action:
As an unprivileged user, execute the following commands:

```
user1>chmod 700 ifs_test
user1>ifs_test
```

Expected Results:
Output other than "Your shell appears well behaved" indicates that the IFS variable does not get reset and under no condition should SUID or SGID scripts be used.

Comments:
SUID and SGID scripts should NEVER be used.

Topic: **FILE SYS SEC**
SubTopic: **Expreserve**

Objective 83
Verify that the expreserve executable is secure.

Rationale:

DII COE SRS Requirement:
Test Actions:
Step: 1

Required Action:
Type in the following command:

```
#ls -l /usr/lib/expreserve
```

Check to see if /usr/lib/ex3.7preserve is setuid root. If not, the following procedure won't work:

a. cd into to your home directory.

b. Create a file called "bin" containing the following lines:

```
IFS=‘ ‘
#  (IFS= should be followed by a single space then return)
cp /bin/sh /the/path/to/your/home/directory/xyzzy
chmod 4755 xyzzy
```

c. After saving the file (and exiting the editor);

Type:
```
%  chmod 755 bin
%  /bin/sh
```

d. From this Bourne shell, type:

```
IFS=/ vi
```

e. You should be in vi. Type "a" (return) and then type a couple of lines of random text into the buffer.

f. Type: `^] (Escape)` :preserve

g. Next exit the editor:

```
:wq
```

10. Enter the command:
```
%  ls -l xyzzy
```

Expected Results:

The date shown for the file is after July 1993.

There should not be a setuid root Bourne shell in your home directory. If the ex command ":preserve" fails, instead you can run a shell from with in vi with the command ":shell", from the shell get the pid of the editor and kill it with a hangup signal.

Topic: **FILE SYS SEC**
SubTopic: **Expreserve**

Comments:
Removal of executable permission will protect the system from this vulnerability, but will also mean that users who edit their files with either vi(1) or ex(1) and have their sessions interrupted, will not be able to recover their lost work. If you implement the above workaround, please advise your users to regularly save their editing sessions.

Topic: **FILE SYS SEC**
SubTopic: **Path**

Objective 87
Verify root's search path is correct.

Rationale:
A search path should never contain the current directory. This is especially true of the superuser account. More generally, a search path should never include a directory that is writeable by other users.

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
As root, execute the following command:

```
echo $PATH
```

OR

review the root search path found in the /.profile, /.cshrc, and /.login files.

Expected Results:
Root's search path does not include the current directory (specified by a ".").

Comments:

Step: 2

Required Action:
As root, execute the following command:

```
ls -ldb `echo $PATH | sed 's/:/ /g'`
```

Expected Results:
None of the directories in the search path should be world writeable.

Comments:

Topic: **FILE SYS SEC**
SubTopic: **Permissions**

Objective 66
Ensure the file systems are configured correctly and securely.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Execute the following command and then review the df.out file

```
#  df -t
```

Expected Results:
The file system should be appropriately partitioned so that no filesystem is approaching 100% full.

Comments:

Topic: **FILE SYS SEC**

SubTopic: **Permissions**

Objective 70

Verify root's startup files are only writeable by root.

Rationale:

Various programs have methods of automatic initialization to set options and variables for the user. All startup files should be protected so only the user can write to them. It is particularly important that the startup files the superuser uses files that are not writeable by others.

DII COE SRS Requirement:

Test Actions:

Step: 1

Required Action:

As root, execute the following commands from root's home directory and verify from the output that the files listed are writeable only by root:

```
#ls -ldb /.login
#ls -ldb /.profile
#ls -ldb /etc/profile
#ls -ldb /.cshrc
#ls -ldb /.kshrc
#ls -ldb /.emacs
#ls -ldb /.exrc
#ls -ldb /.forward
#ls -ldb /.rhosts
#ls -ldb /.dtprofile
#ls -ldb /.Xdefaults
```

Expected Results:

Permissions of existing files is 600 or 400 and are owned by root.

Comments:

Depending on the system configuration, all of the files listed in "Required Actions" may not exist.

Step: 2

Required Action:

As root, execute the following commands from root's home directory:

```
#/usr/ucb/more  /.login
#/usr/ucb/more  /.profile
#/usr/ucbmore  /etc/profile
#/usr/ucb/more  /.cshrc
#/usr/ucb/more  /.kshrc
#/usr/ucb/more /.emacs
#/usr/ucb/more  /.exrc
#/usr/ucb/more  /.forward
#/usr/ucb/more  /.dtprofile
```

```
#/usr/ucb/more /.Xdefaults
```

and on any executable that is referenced in the file being viewed execute the command:

```
ls -ldb
```

Topic: **FILE SYS SEC**
SubTopic: **Permissions**

Expected Results:
Permissions of all files referenced in the listed files are 600 or 400 and are owned by root.
Comments:

Topic: **FILE SYS SEC**
SubTopic: **Permissions**

Objective 72
Verify all root executable files are owned by root and are not world or group writeable.

Rationale:
System Administrators should be trained to type in full pathname of files to be executed and to ensure that any executable that is not located in a protected directory is safe to execute.

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
Type in the following commands:

```
#ls -lgdb / /bin /etc /usr /usr/bin /usr/etc

#ls -lgb /bin /etc /usr/bin /usr/etc
```

Expected Results:
Listed files are owned by root and are not world or group writeable.

Comments:
All executables run by root should be owned by root and all executables run by root should not be world or group writeable and all executables run by root should be located in a directory where every directory in the path is owned by root and is not group or world writeable. In particular, the following directories should not be group or world writeable: /, /bin, /etc, /usr/bin, /usr/etc. System Administrators should be trained to type in full pathname of files to be executed and to ensure that any executable that is not located in the protected directories listed above are safe to execute.

Topic: **FILE SYS SEC**
SubTopic: **Permissions**

Objective 78
Identify all world-writeable files on the system and verify their need for world-write access.

Rationale:
World-writeable files, directories, and devices represent a potential security hole in a system. It is important to periodically identify them and verify the need for world-write access. Notable files that may be world-writeable include: /tmp, /usr/tmp, and /dev/tty*(Garfinkel and Spafford, 1992).

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
As root, execute the following commands:

```
# /bin/find / -type f \( -perm -2 -o -perm -20 \) -exec ls -lgb {} \;

# /bin/find / -type d \( -perm -2 -o -perm -20 \) -exec ls -lgdb {} \;
```

Expected Results:
There are no unexpected world writeable files or directories on your system. Files should be world-writeable only if there is a legitimate requirement.

Comments:
The following files and directories may safely remain world-writeable:

/tmp and contents
/var/tmp and contents
/var/preserve
/var/mail
(and many more...)

COPS, Tiger, SPI all provide checking of file permissions.

Topic: **FILE SYS SEC**
SubTopic: **Permissions**

Objective 79
Verify that all world-readable, but not world or group writeable, non-setuid/setgid system files and directories are owned by root. (see rationale)

Rationale:
Many systems ship files and directories owned by bin (or sys). This varies from system to system and may have serious security implications.

CHANGE all non-setuid files and all non-setgid files and directories that are world readable but not world or group writeable and that are owned by bin to ownership of root, with group id 0 (wheel group under SunOS 4.1.x).

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
As root, execute the following command:

```
/usr/bin/find / -perm -4 ! \( -perm -6022 \)  \
       \( -type f  -o -type d \) \
       ! -user root -group 0 -exec ls -lgdb {} \;
```

Expected Results:
Any output from this command indicates a file or directory that does not meet the criteria listed in the rationale and should be investigated carefully.

Comments:
Use of a tool such as Tiger, COPS, or SPI would be very useful and save work.

Topic: **FILE SYS SEC**
SubTopic: **Permissions**

Objective 82
Verify the startup and shutdown scripts are valid and protected.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
As root, execute the following command:

```
/bin/find /etc \( -perm -2 -o -perm -20 \) -exec ls -ld {}
```

Expected Results:
There should be no output indicating that the /etc directory and its contents are not group or world writeable.

Comments:

Step: 2
Required Action:
Review all startup and shutdown scripts and configuration files. These scripts are located in the /etc directory and all begin with rc.

Expected Results:
Any task performed in the startup script is performed securely. Any service started or task performed is approved. Any directory that contains a script, executable, or configuration file that is executed in the rc scripts during bootup and shutdown is not writeable by a user other than root.

Comments:
Work intensive!

Topic: **FILE SYS SEC**
SubTopic: **Permissions**

Objective 85
Identify the SUID and SGID files on the system and verify their need for SUID and SGID privilege.

Rationale:
SUID and SGID files allow an unprivileged user to accomplish tasks that require privileges. When a SUID program is run, its effective UID becomes that of the user who created the program, rather than the user who is running it. When a SGID program runs, its effect GID becomes that of the creating user.

Shell scripts that have the setuid or setgid bits set on them are not secure, regardless of how many safeguards are taken when writing them. Setuid and setgid shell scripts should never be allowed on any UNIX system (Curry, 1990).

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
As root, execute the following command:

```
#/bin/find / -type f \( -perm -4000 -o -perm -2000 \) \
 -exec ls -lgdb {} \;
```

Expected Results:
Verify that all of the programs listed as output should be SUID or SGID. Only authorized files should be SUID or SGID.

Comments:
Depending on the system configuration, the output may be lengthy and it may be easier to review if piped to an output file, which can then be printed and reviewed. Tools such as COPS, Tiger, and SPI report SUID and SGID programs.

Topic: **FILE SYS SEC**
SubTopic: **Permissions**

Objective 86
Determine if users can "give away" files, and if so, if they can "give away" an SUID file to root.

Rationale:
The last defense against system crackers are the permissions offered by the file system. Each file or directory has three sets of permission bits associated with it: one set for the user who owns the file, one set for the users in the group with which the file is associated, and one set for all other users (the "world" permissions). Each set contains three identical permission bits, which control the following (Curry, 1990):

read - If set, the file or directory may be read. In the case of a directory, read access allows a user to see the contents of a directory (the names of the files contained therein), but not to access them.

write - If set, the file or directory may be written (modified). In the case of a directory, write permission implies the ability to create, delete, and rename files. Note that the ability to remove a file is not controlled by the permissions on the file, but rather the permissions on the directory containing the file.

execute - If set, the file or directory may be executed (searched). In the case of a directory, execute permission implies the ability to access files contained in that directory.

In addition, a fourth permission bit is available in each set of permissions. This bit has a different meaning in each set of permission bits:

setuid - If set in the owner permissions, this bit controls the "set user id" (setuid) status of a file. Setuid status means that when a program is executed, it executes with the permissions of the user owning the program, in addition to the permission of the user executing the program. This bit is meaningless on nonexecutable files.

setgid - If set in the group permissions, this bit controls the "set group id" (setgid) status of a file. This behaves in exactly the same way as the setuid bit, except that the group id is affected instead. This bit is meaningless on non-executable files (but see below).

sticky - If set in the world permissions, the "sticky" bit tells the operating system to do special things with the text image of an executable file. It is mostly a hold-over from older versions of UNIX, and has little if any use today. This bit is also meaningless on nonexecutable files (but see below).

Under some versions of UNIX, users can run the chown command to change the ownership of a file that they own to that of any other user on the system, allowing them to "give away the file."

DII COE SRS Requirement:
3.2.4.4 The COE shall provide controls to limit the propagation of access rights.

Topic: FILE SYS SEC
SubTopic:Permissions

Test Actions:

Step: 1

Required Action:

As an unprivileged user, execute the following commands:

```
%touch test
%ls -lg test
%chgrp root test
%ls -lg test
%chmod 2755 test
%ls -lg test
%chown root test
%ls -lg test
```

Expected Results:

Each attempt to change the group to root should result in an error message of "Permission denied". Output should be similar to the following:

```
user1>touch test
user1>ls -lg test
-rw-------  1 mls      rg021           0 Oct 21 09:41 test
user1>chgrp root test
chgrp: test: Not owner
user1>ls -lg test
-rw-------  1 mls      rg021           0 Oct 21 09:41 test
user1>chmod 2755 test
user1>ls -lg test
-rwxr-sr-x  1 mls      rg021           0 Oct 21 09:41 test
user1>chgrp root test
chgrp: test: Not owner
user1>ls -lg test
-rwxr-sr-x  1 mls      rg021           0 Oct 21 09:41 test
user1> rm test
```

Comments:

A general user should not be able to change the group of an SUID or SGID file (or any file) to any other group especially root.

Step: 2

Required Action:

As an unprivileged user, execute the following commands:

```
%touch test
%ls -lg test
%chown root test
```

```
%ls -lg test
%chmod 4755 test
%ls -lg test
%chown root test
%ls -lg test
```

Topic: **FILE SYS SEC**
SubTopic **Permissions**

Expected Results:
Each attempt to change the owner to root should result in an error message of "Permission denied". Output should be similar to the following:

```
user>touch test
user>ls -lg test
-rw-------  1 mls      rg021           0 Oct 21 09:30 test
user>chown root test
chown: test: Not owner
user>chmod 4755 test
user>ls -lg test
-rwsr-xr-x  1 mls      rg021           0 Oct 21 09:30 test*
user>chown root test
chown: test: Not owner
user>ls -lg test
-rwsr-xr-x  1 mls      rg021           0 Oct 21 09:30 test*
user>rm test
user>
```

Comments
A general user should not be able to change the ownership of an SUID or SGID file (or any file) to any other user especially root.

Topic: **FILE SYS SEC**
SubTopic: **Unauthorized Device Files**

Objective 75
Ensure no unauthorized device files are present on the system.

Rationale:
The system's disks should be periodically scanned for unauthorized device files.

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
As root, execute the following command:

```
#/bin/find / \( -type c -o -type b \) -exec ls -lgdb {} \; \
| grep -v "/dev/"
```

Expected Results:
There should be no output from this command indicating that there are no device files outside the /dev

Comments:
Any device outside the /dev directory should be viewed with GREAT suspicion.

NOTE: ncheck locates SUID files also. The -s parameter of the ncheck command displays special files and files with set-user-ID mode. This parameter can be used to discover concealed violations of security policy. The ncheck command would be run as root and executed as follows:

#/etc/ncheck -s

Step: 2

Required Action:
As root, execute the following command:

```
/bin/find /dev ! \( -type l -o -type c -o -type b \) -exec ls -lgdb {}
\;
```

Expected Results:
All files in /dev and /devices are special files.

Comments:

Step: 3

Required Action:
As root, execute the following command:

```
#/bin/find / \( -type c -o -type b \) ! -user root -exec ls -ldb {} \;
```

Expected Results:
There are no special device files owned by root that should not be owned by root.

Comments:
Any device outside the /dev and /devices directory not owned by root should be viewed with even GREATER suspicion.

Topic: FINGER
SubTopic: Penetrate

Objective 130
Determine if finger and fingerd are enabled on the system. If enabled, verify Finger is securely configured.

Rationale:
The "finger" service, provided by the finger program, allows you to obtain information about a user such as her full name, home directory, last login time, and in some cases when she last received mail and/or read her mail. The fingerd program allows users on remote hosts to obtain this information (Curry, 1990).

A bug in fingerd was also exercised with success by the Internet worm. If your version of fingerd is older than November 5, 1988, it should be replaced with a newer version (Curry, 1990).

The finger program has two uses: If finger is run with no arguments, the program prints the username, full name, location, login time, and office telephone number of every user currently logged into the local system. If finger is run with a name argument, the program searches through the /etc/passwd file and prints detailed information for every user with a first, last, or user name that matches the name you specified. finger makes it easy for intruders to get a list of the users on the system.

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Type in the following command:

```
user1>finger root@localhost
```

Expected Results:
Error message indicates that the finger daemon is not enabled. Output of information regarding root indicates that finger is enabled.

Comments:
Finger should NOT be enabled unless there is a legitimate need for it.
Related services that should be considered for removal are systat and netstat.

Step: 2
Required Action:
Execute the following command:

```
user1>finger 23234123123123123@localhost
```

Expected Results:
Only login information on users currently logged on the system are provided.

Comments:
There is a bug in some operating systems which allows a remote finger request to dump all known user finger profiles back out to the requester.

Topic: **FTP**
SubTopic:

Objective 132
Determine whether FTP is enabled on the system. If FTP is enabled, verify that it has been securely configured.

Rationale:
The File Transfer Protocol (FTP) allows the user to transfer complete files between systems. ftp is the client program; /etc/ftpd is the server.

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Attempt to login via FTP into the workstation:

Type the following commands:

```
%  ftp <hostname>
name (localhost:idname):  <valid username>
```

Expected Results:
If login succeeds than ftp is enabled.

Comments:

Topic: **FTP**
SubTopic:

Objective 136
Verify the FTP users file contains the appropriate accounts.

Rationale:
The /etc/ftpusers file contains a list of the users who are not allowed to use FTP to access any files. This file should contain all accounts that are not used by actual users.

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
Type the following commands:

```
ls -lg /etc/ftpusers
more /etc/ftpusers
```

Expected Results:
The permissions do not allow group/world write and the file is owned by root. Typical accounts that should be included are uucp, news, bin, ingress, news, nobody, daemon, and root.

Comments:
The ftpusers file should contain a list of users who are not allowed access to the system using the File Transfer Protocol (FTP). If this file is missing, the list of users is considered to be empty, so that any user may use FTP to access the system if the other criteria for access are met.

Topic: **FTP**
SubTopic:

Objective 138
Determine whether Trivial FTP is enabled on the system and if enabled, verify that it has been securely configured.

Rationale:
The TFTP is used to allow diskless hosts to boot from the network. Basically, TFTP is a stripped-down version of FTP - there is no user authentication. Because they are so stripped-down, many implementations of TFTP have security holes
(Curry, 1990).

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
As an unprivileged user execute the following commands:

```
%  tftp
tftp>  connect localhost
tftp>  get /etc/passwd testfile
tftp>  quit
%ls -l testfile
%more testfile
%rm testfile
```

Expected Results:
If tftp does not respond with "Error Code 2: Access Violation" and instead transfers the file, the version of tftp should be replaced with a newer one.

Comments:
The use of tftp does not require an account or password on the remote system. The -s option ensures that tftpd will only start with the home and root directory set to /tftpboot.

Topic: **FTP**
SubTopic:

Objective 140
Verify that Trivial FTP does not run with privileges.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Type the following command:

```
% ls -lgdb /usr/bin/tftp
```

Expected Results:
Verify that the file is not running SUID or SGID. The tftp file should not have the SUID or SGID bits set.

Comments:

Topic: **FTP**
SubTopic: **Anonymous FTP**

Objective 134
Determine whether anonymous FTP is enabled on the system. If anonymous FTP is enabled, verify that it has been securely configured.

Rationale:
Anonymous FTP allows users who do not have an account on a machine to have restricted access in order to transfer from a specific directory. Because the anonymous FTP feature allows anyone to access the system (albeit in a very limited way), it should not be made available on every host on the network. If anonymous ftp is required, one machine should be chosen (preferably a server or standalone host) on which to allow this service.

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
To ascertain whether you are running anonymous ftp, try to connect to the localhost using anonymous ftp. Be sure to give an RFC822-compliant username (e.g., mcguire@ncr.disa.mil) as the password. Type the following commands to ascertain whether anonymous ftp is enabled:

```
%  ftp <hostname>
name (localhost:idname):  anonymous
```

Expected Results:
If the error message "530 User anonymous unknown" (or similar error) is returned then anonymous ftp is disabled. If the system instead replies with the string "331 Guest login ok" (or similar message) and then prompts for a password, anonymous ftp access is enabled.

Comments:
Anonymous ftp should not be enabled unless there is a legitimate business need.

Step: 2
Required Action:
If anonymous FTP is enabled, verify that the ftp account has been created and has been disabled by placing an asterisk (*) in the password field. Verify that the account has been given a special home directory, such as /usr/ftp or /usr/spool/ftp.

Expected Results:

Comments:

Step: 3
Required Action:
If anonymous FTP is enabled, verify that the ftp owns its home directory and that it is unwriteable by

anyone.

Expected Results:

Comments:

Topic: FTP
SubTopic:: Anonymous FTP

Step: 4

Required Action:
If anonymous FTP is enabled, verify that the directory ftp/bin is owned by the super-user and unwriteable by anyone. Verify that a copy of the ls program is in this directory.

Expected Results:

Comments

Step: 5

Required Action:
If anonymous FTP is enabled, verify that the directory ftp/etc is owned by the super-user and unwriteable by anyone. Verify that copies of the password and group files are in this directory, with all the password fields changed to asterisks (*). Note: The only account that must be present is "ftp."

Expected Results:

Comments:

Step: 6

Required Action:
If anonymous FTP is enabled, verify that the directory ftp/pub is owned by "ftp" and world-writeable.

Expected Results:

Comments:

Topic: **I&A**
SubTopic: **Accounts**

Objective 98
Verify there are no accounts on the system that have not been used within a reasonable amount of time.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Type in and run the following script to determine which users have not logged in within the last month:

```
#!/bin/sh
date
uname -a
PATH=/bin:/usr/bin;export PATH
umask 077
THIS_MONTH=`date | awk '{print $2}'`
/bin/last | /bin/grep $THIS_MONTH | awk '{print $1}' | sort -u >
users1$$
cat /etc/passwd | /bin/awk -F: '{print $1}' | /bin/sort -u > users2$$
/bin/comm -13 users[12]$$
/bin/rm -f users[12]$$
```

Expected Results:
No USER login account names should be returned. If any user names are returned these should be considered dormant accounts and should be disabled or deleted.

Comments:

Topic: **I&A**
SubTopic: **Accounts**

Objective 100
Verify there are no duplicate GIDs.

Rationale:
The UNIX operating system relies on the GID to identify groups. It does not rely on the group name. Therefore, if two different group names have the same GID they are indistinguishable to the operating system.

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
Review the file /etc/group file.

Expected Results:
There should not be duplicate GIDs.

Comments:
Group ids must be distinct integers between 0 and 32,767. If the environment is networked, users should have the same unique UID across the entire network. GID 0 is generally reserved for the groups "root" or "wheel" and GID 1 is reserved for the group "daemon".

Topic: **I&A**
SubTopic: **Password Management**

Objective 105
Verify the system enforces individual user accountability, a globally-unique valid userid and password is required for all users to access the system, and the user's identity is associated with all auditable actions performed.

Rationale:
Some sites have installed accounts with names such as "who," "date," "lpq," and so on that execute simple commands. These accounts are intended to allow users to execute these commands without having to log in to the machine. Typically these accounts have no password associated with them, and can thus be used by anyone. Many of the accounts are given a user id of zero, so that they execute with super-user permissions (Curry, 1990).

The problem with these accounts is that they open potential security holes. By not having passwords on them, and by having super-user permissions, these accounts practically invite crackers to try to penetrate them. Usually, if the cracker can gain access to the system, penetrating these accounts is simple, because each account executes a different command. If the cracker can replace any one of these commands with one of his own, he can then use the unprotected account to execute his program with super-user permissions (Curry, 1990).

Simply put, accounts without passwords should not be allowed on any UNIX system (Curry, 1990).

An account without a password is an easy target for an intruder and subjects the entire system to risk.

DII COE SRS Requirement:
3.2.1.1 The COE shall enforce individual accountability by providing the capability to uniquely identify each individual system user.
3.2.1.1.1 The COE shall require users to identify themselves before beginning to perform any actions that the system is expected to mediate.
3.2.1.2 Each user shall be identified by a globally unique user name or userID that will follow a standard set of processes or rules for formation.
3.2.1.3 The COE shall provide the capability of associating the user's identity with all auditable actions taken by that individual.

Test Actions:
Step: 1
Required Action:
Type the following command:

```
%  awk -F: 'length ($2)<1 {print $1}' < /etc/passwd
```

Expected Results:
There should be no output, indicating that all accounts have passwords. Account names that do not require passwords will be printed and should be deleted, locked, or password protected.

Comments:

Topic: **I&A**
SubTopic: **Password Management**

Objective 106
Verify the installation-provided userIDs do not have default passwords.

Rationale:
Several accounts come with a UNIX computer system. (These accounts are normally at the beginning of the /etc/passwd file and have names like bin, lib, uucp, and news.) Either disable these accounts or change their passwords (Garfinkel and Spafford, 1992).

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Attempt to log in using each of the following IDs with its default password:

ID	Password
guest	guest
root	root
system	manager

Expected Results:
The default passwords should not be valid for the accounts.

Comments:
After installation be sure to change all default passwords, lock the account, or delete the account.

Note: COPS, Tiger, and SPI check for common default passwords.

Topic: **I&A**
SubTopic: **Password Management**

Objective 112
Verify password life is limited to a maximum of 180 days and the user is notified prior to password expiration.

Rationale:
Some UNIX systems allow the system administrator to set a "lifetime" for passwords. Users whose passwords are older than the time allowed are forced to change their passwords the next time they log in. If a user's password is exceptionally old, the system may prevent the user from logging in altogether (Garfinkel and Spafford, 1992).

DII COE SRS Requirement:
3.2.1.4.2 Password life shall be limited to a maximum of 180 days. The COE shall notify the user prior to password expiration.

Test Actions:
Step: 1

Required Action:
Interview the ISSO. Browse the /etc/passwd file and determine if the fourth field is blank.

Expected Results:
Password aging should be enabled.

Comments:

Topic: **I&A**
SubTopic: **Privileged Accounts**

Objective 101
Verify only authorized users are members of the “wheel” or “root” group.

Rationale:
By convention, the “wheel” or “root” group is the list of all of the computer's system administrators. In some versions of UNIX, a user who is not in the wheel group cannot use the su command to become superuser. In some versions of UNIX, people in the wheel group can provide their own password to su instead of the superuser password to become root.

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
Browse the /etc/group file. Verify that only authorized usernames appear with the root group name.

Expected Results:
Only authorized usernames should appear with the root group name.

Comments:
GID 0 is generally reserved for the groups "root" or "wheel" therefore, only privileged users should be members of this group.

Topic: **I&A**
SubTopic: **Accounts**

Objective 99
Verify there are no duplicate UIDs.

Rationale:
The UNIX operating system relies on the UID to identify accounts. It does not rely on the account name. Therefore, if two different account names have the same UID they are indistinguishable to the operating system.

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
Review the file /etc/passwd.

Expected Results:
There should not be duplicate UIDs. If there are duplicate UIDs, the accounts should be disabled.

Comments:
User ids must be distinct integers between 0 and 32,767. If the environment is networked, users should have the same unique UID across the entire network. Root uses UID 0, Bin uses UID 1, and Daemon uses UID 2. In addition, it is customary to use the lower UIDs for non-human logins (i.e., UUCP). It is not recommended to re-use UIDs after a user account is deleted.

Topic: **I&A**
SubTopic: **Accounts**

Objective 103
Verify site identifying information is stored for all user accounts on the system.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
Browse the /etc/passwd file.

Expected Results:
The fifth field of the file should be filled in with relevant data (i.e., full user name and user location).

Comments:

Topic: **I&A**
SubTopic: **Accounts**

Objective 102
Verify there are no guest accounts on the system.

Rationale:
Guest accounts present a security hole. By their nature, these accounts are rarely used, some are always used by people who should only have access to the machine for the short period of time that they are guests. The most secure way to handle guest accounts is to install them on an as-needed basis, and delete them as soon as the people using them leave. Guest accounts should never be given simple passwords such as "guest" or "visitor," and should never be allowed to remain in the password file when they are not being used (Curry, 1990).

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Browse the /etc/passwd file to determine if there is a guest account. If so, try simple passwords such as "guest" and "visitor".

Expected Results:
Guest accounts should not exist. If they do they should NOT have trivial passwords.

Comments:

Topic: **I&A**
SubTopic: **Password Management**

Objective 71
Ensure authentication data is protected from being accessed by unauthorized users.

Rationale:
It is no longer considered secure to place even encrypted passwords in the world-readable /etc/passwd file. As a result, numerous vendors have introduced shadow password files. These files have the same encrypted passwords, but the passwords are stored in special files that cannot be read by most users on the system (Garfinkel and Spafford, 1992).

DII COE SRS Requirement:
3.2.1.5 The COE shall protect authentication data from being accessed by unauthorized users.

Test Actions:
Step: 1

Required Action:
Browse the /etc/passwd file.

Expected Results:
If no encrypted passwords are present, that is, if all password fields contain an * or -, then the shadow password file has been implemented.

Comments:

Topic: **MAIL**
SubTopic:

Objective 32
Verify the "decode" and "uudecode" aliases have been removed from the aliases file (/etc/aliases or /usr/lib/aliases).

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Type the following command:

```
#vi /usr/lib/aliases
```

Search for decode by typing "/decode" and press return.

Expected Results:
A message should be printed to the bottom of the window as follows:

```
Pattern not found
```

OR the decode alias line appears as follows:

```
#decode:  "|/usr/bin/uudecode"
```

Comments:
After modifying the /etc/aliases file the /etc/newaliases executable must be executed.

Topic: **MAIL**
SubTopic: **Penetration Test**

Objective 33
Verify sendmail does not support the wiz command.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Type in the following commands:

```
%telnet localhost 25
wiz
quit
```

Expected Results:
Sendmail should respond to the wiz command with "500 Command unrecognized." If sendmail responds to the wiz command in any other way, then the system is vulnerable to attack and sendmail should be replaced.

The session should appear similar to the following:

```
user>telnet localhost 25
Trying 127.0.0.1 ...
Connected to localhost.
Escape character is '^]'.
220 ziggy. Sendmail 5.x/SMI-SVR4 ready at Fri, 18 Oct 1996 15:48:03 -
0400
wiz
500 Command unrecognized
quit
221 ziggysol24. closing connection
Connection closed by foreign host.
user>
```

Comments:
The wiz command should be disabled.

Topic: **MAIL**
SubTopic: **Penetration Test**

Objective 34
Verify sendmail does not support the debug command.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Type in the following commands:

```
%  telnet localhost 25
debug
quit
```

Expected Results:
Sendmail should respond to the debug command with "500 Command unrecognized." If sendmail responds to the debug command in any other way, then the system is vulnerable to attack and sendmail should be replaced.

The session should appear similar to the following:

```
user>telnet localhost 25
Trying 127.0.0.1 ...
Connected to localhost.
Escape character is '^]'.
220 ziggy. Sendmail 5.x/SMI-SVR4 ready at Fri, 18 Oct 1996 15:48:03 -
0400
debug
500 Command unrecognized
quit
221 ziggysol24. closing connection
Connection closed by foreign host.
user>
```

Comments:
The debug command should be disabled.

Topic: **MAIL**
SubTopic: **Penetration Test**

Objective 35
Verify sendmail does not support the kill command.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Type in the following commands:

```
%telnet localhost 25
kill
quit
```

Expected Results:
Sendmail should respond to the kill command with "500 Command unrecognized." If sendmail responds to the kill command in any other way, then the system is vulnerable to attack and sendmail should be replaced.

The session should appear similar to the following:

```
user>telnet localhost 25
Trying 127.0.0.1 ...
Connected to localhost.
Escape character is '^]'.
220 ziggy. Sendmail 5.x/SMI-SVR4 ready at Fri, 18 Oct 1996 15:48:03 -
0400
kill
500 Command unrecognized
quit
221 ziggysol24. closing connection
Connection closed by foreign host.
user>
```

Comments:
The kill command should be disabled.

Topic: **MAIL**
SubTopic: **Sendmail**

Objective 31
Verify sendmail is configured correctly.

Rationale:
Electronic mail is one of the main reasons for connecting to outside networks. On most versions of Berkeley-derived UNIX systems, including those from Sun, the sendmail program is used to enable the receipt and delivery of mail. Because of its design, sendmail runs as the superuser, making its security holes a significant problem for the entire system. As with the FTP software, older versions of sendmail have several bugs that allow security violations. One of these bugs was used with great success by the Internet worm (Curry, 1990).

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
If you use a vendor version of sendmail, ensure that you have installed the latest patches as sendmail(8) has been a source of a number of security vulnerabilities. Refer to AUSCERT Advisories SA-93:10, AA-95.08 and AA-95.09b and CERT Advisories CA-94:12, CA-95:05 and CA-95:08.

Browse the /usr/lib/ sendmail.cf and verify the following lines:

```
Mlocal, P=/bin/mail, F=rlsDFMmnP, S=10, R=20, A=mail -d $u
# Mprog, P=/bin/sh, F=lsDFMeuP, S=10, R=20, A=sh -c $u
Mprog, P=/bin/true, F=lsDFMeuP, S=10, R=20, A=true
```

Expected Results:
Sendmail should be properly configured.

Comments:
Step: 2
Required Action:
Execute the following command:

```
#vi /etc/mail/sendmail.cf
```

Expected Results:
Any line starting with "OW" only has a "*" next to it (Or does not exist).

The options part of the general configuration information section includes lines similar to:

```
    # log level
    OL9
```

OR (for sendmail 8.7 or later)

```
# log level
O LogLevel=9
```

(The higher the number, the more information is logged)

Topic: **MAIL**
SubTopic: **Sendmail**

The Local and Program Mailer specification section contains a commented out Mprog entry similar to the following:

```
#Mprog,  P=/bin/sh,   F=lsDFMeuP,  S=10, R=20, A=sh -c $u
```

OR a modified Mprog line similar to the following:

```
#Mprog,  P=/bin/true,   F=lsDFMeuP,  S=10, R=20, A=true
```

Comments:
Sendmail doesn't deliver mail, it invokes the program listed on the Mlocal line in the sendmail.cf file (after setuiding itself to the receiving user). You'll have to check out the capabilities of that program to be sure (although sendmail 8 comes with a binmail delivery program which doesn't do any forwarding).

Step: 3

Required Action:
Type the following command:

```
#vi /etc/mail/mailx.rc
```

Expected Results:
The following lines appear as specified:

```
set append dot
if t
   set SHELL=/bin/true
else
   set SHELL=/bin/true
endif
```

Comments:

Step: 4

Required Action:
As root execute the following command:

```
#find / -name .forward -exec ls -ald {} \; -exec more {} \;
```

Expected Results:
There are no .forward files listed.

Comments:
If the responsible person permits .forward files, any .forward files in user home directories do not execute an unauthorized command or program.

Topic: **MAIL**
SubTopic: **Sendmail**

Step: ***5***

Required Action:
Enter the following command:

```
#vi /etc/syslog.conf
```

Expected Results:
The file syslog.conf contains lines similar to:

```
mail.info       /dev/console
mail.info       /var/adm/message
```

Comments:
These lines cause mail informational messages to be written to the console and to the messages file.

Step: ***6***

Required Action:
Review the file: `/usr/lib/aliases`.

Expected Results:

MAILER-DAEMON should be redirected to Postmaster. audit_warn should be redirected to the system administrator's account. Nobody should be redirected to /dev/null.

The "decode" alias should be commented out by placing a "#" at the beginning of the line. For this change to take effect you will need to run /usr/bin/newaliases. If you run NIS, you will then need to rebuild your maps:

```
# cd /var/yp; /usr/bin/make aliases
```

Ensure that all programs executable by an alias are owned by root, have permissions 755 and are stored in a systems directory (e.g., /usr/local/bin).

Comments:
The white space between the syslog.conf entries must be a tab character.

Topic: **MAIL**
SubTopic: **Sendmail bug Penetration Tests**

Objective 274
Verify that the sendmail -d bug does not exist.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
From a command shell, execute the following command:

```
# sendmail -d3294967296
```

Expected Results:
This command does not cause a segmentation fault.

Comments:
On some versions of sendmail it is possible to get root access by supplying greater than normal address space ranges that are used in its array index to the -d flag. If this causes a segmentation fault then you'll likely have a bug in your version of sendmail. The problem is that numbers in this range may skip the range checks and result in accessing negative indexes into the debug array. Hence it is possible to write to locations in memory before the debug array.

Topic: **Markings**
SubTopic:

Objective 6
Verify a security warning is displayed prior to the login process indicating restrictions that apply to logins, the highest classification of information processed on the system, and that misuse is subject to applicable penalties.

Rationale:

DII COE SRS Requirement:
3.2.7.1 The COE shall display a security warning prior to the login process that indicates the highest classification of information processed on the system and that misuse is subject to applicable penalties.

Test Actions:
Step: 1

Required Action:
Prior to login view the monitor. Review the /etc/motd file and verify that the text in the file contains the text that is the site approved warning to users logging on the system.

Expected Results:
A security warning is displayed prior to the login process indicating restrictions that apply to logins, the highest classification of information processed on the system, and that misuse is subject to applicable penalties.

Comments:
DII COE does not use /etc/motd.

Topic: **NETWORK CONFIGURATION**
SubTopic:

Objective 41
Verify the network services are appropriately configured and defined.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Verify that the network services are appropriately configured by browsing /etc/inetd.conf using the following command:

```
#/usr/bin/vi   /etc/inetd.conf
```

Expected Results:
Unnecessary network services should be disabled.

A "#" starts each line identifying a disabled service. Verify that the following services are disabled:

name, shell, login, exec, comsat, talk, uucp, finger,
systat, netstat, admind, rquotad, rusersd, sprayd,
walld, rstatd, rexd, rpc.cmsd, rpc.ttdbserverd

Comments:
The following services should be disabled if possible!

name - obsolete name server protocol
shell - allows remote user via rsh to run processes on this system
login - allows remote user via rlogin
exec - allows remote users access via rexec
comsat - real-time intrusive notification to users that mail has arrived
talk - remote chat protocol
uucp - UNIX-to-UNIX copy over TCP
finger - remote access to local user information
systat - allows remote users to view the process table
netstat - allows remote users to view the list of active network connections
admind - allows remote users to execute remote administrative activities
rquotad - provides disk quota information to NFS clients
rusersd - provides local user information
sprayd - allows remote users to send a stream of IP packets to the host and have them acknowledged
walld - allows remote users to post messages to system users
rstatd - allows remote users to view system information such as load
rexd - obsolete remote execution server with no security

rpc.cmsd - calendar manager
rpc.ttdbserverd - tool talk database server that allows object linking. MAY BE NEEDED for DCE.

Topic: NETWORK CONFIGURATION

SubTopic:

Step: 2

Required Action:
Portmapper

Disable any non-required network services that are started up in the system startup procedures and register with the portmapper. The following command determines which services are registered with the Portmapper:

```
# /user/etc/rpcinfo -p localhost
```

Expected Results:

Unrequired network services are disabled.

Comments:

Step: 3

Required Action:
Verify that only appropriate network services are defined.

Review: `/etc/services`

Expected Results:
The services database lists the names of TCP and UDP services and their port numbers. It is used by programs that call network services.

Comments:

Topic: **NETWORK CONFIGURATION**
SubTopic:

Objective 43
Verify netrc files are not used.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
As root, execute the following command:

```
#/bin/find / -name .netrc -exec ls -ld {} \; -exec more {} \;
```

Expected Results:
There should NOT be any output from this command. Any output indicates the existence of a .netrc file on the system. The file path, permissions and contents are listed.

Comments:
The .netrc file should not exist on a secure system.
If the responsible security officer has approved the use of .netrc files for a specific purpose:
Do not store password information in .netrc files.
Set permissions on .netrc files to disallow read and write access by group and world (i.e. 600).

Topic: **NETWORK CONFIGURATION**
SubTopic:

Objective 113
Verify Subnet addresses are appropriately configured.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
Review: `/etc/netmasks`.

Expected Results:
The correct subnet definitions must be obtained from the local network administrator.

Comments:

Topic: **NETWORK CONFIGURATION**
SubTopic: **.rhost files**

Objective 115
Determine if any rhost files are used on the system.

Rationale:
The .rhosts file is similar in concept and format to the hosts.equiv file, but allows trusted access only to specific host-user combinations, rather than to hosts in general. Each user may create a .rhosts file in his home directory, and allow access to his account without a password. Most people use this mechanism to allow trusted access between accounts they have on systems owned by different organizations that do not trust each other's hosts in hosts.equiv. Unfortunately, this file presents a major security problem: While hosts.equiv is under the system administrator's control and can be managed effectively, any user may create a .rhosts file granting access to whomever he chooses, without the system administrator's knowledge (Curry, 1990).

The only secure way to manage .rhosts files is to completely disallow them on the system. The system administrator should check the system often for violations of this policy (Curry, 1990).

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
As root, execute the following command:

```
#/bin/find / -name .rhosts -exec ls -ldb {} \; -exec more {} \;
```

Expected Results:
There should be no output from this command. Output means that a .rhosts file has been found. Users should not have a .rhosts file.

Comments:
Cron should be used to periodically check for, report the contents of, and remove .rhosts files.

If there is a genuine need for .rhosts files (e.g., running backups over a network unattended) and their use has been approved by responsible security officer:

the first character of any .rhosts file is not '-'.

The permissions of all .rhosts files are set to 600.

The owner of each .rhosts file is the account's owner.

No .rhosts file contains the symbol "+" on any line.

Usage of netgroups within .rhosts does not allow unintended access to this account

.rhosts files do not use '!' or '#'.

Topic: **NETWORK CONFIGURATION**
SubTopic: **NFS**

Objective 76
Verify the files on the server are not world-writeable or group-writeable.

Rationale:
Because the NFS server maps root to nobody, you can protect files and directories on your server by setting their owner to root and making them not world-writeable or group-writeable.

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
Browse the /etc/dfs/dfstab file using the following command:

```
#vi /etc/exports
```

and for each shared filesystem run the following command:

```
/bin/find filesystem \( -perm -2 -o -perm -20 \) -exec ls -ldg {} \;
```

Expected Results:
No files should be listed.

Comments:

Topic: **NETWORK CONFIGURATION**
SubTopic: **NFS**

Objective 77
Ensure filesystems are mounted with the nosuid option and read-only where practical. If read-only is not practical, verify system files and user home directories are not mounted.

Rationale:
In some versions of UNIX, it is possible to turn off the SUID and SGID bits on mounted filesystems by specifying the nosuid option with the mount command. If available, this option should always be specified when a filesystem is mounted unless there is an overriding reason to import SUID or SGID files from the mounted filesystem (Garfinkel and Spafford, 1992).

One of the best ways to protect sensitive files and directories is to mount them on read-only disks. It is recommended that the following directories be mounted as read-only partitions: /, /usr/bin, /bin, /etc, /lib, /usr/lib, /usr/ucb (if it exists), /usr/include, /usr/src, /usr/etc (if it exists) (Garfinkel and Spafford, 1992).

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Browse the /etc/fstab file using the following command:

```
#vi /etc/fstab
```

Expected Results:
The flag rw should only exist if a legitimate need exists and the flag nosuid should appear.

Comments:
Each nfs entry in the /etc/fstab file should appear similar to the following line:

```
#device        device     mount      FS      fsck     mount       mount
#to mount      to fsck     point     type    pass    at boot
options
Exporthost:/ExportDirPath  -  /mountpoint   nfs  -    yes ro,bg,nosuid
```

OR if mounting the filesystem by from the command line use the following command:

```
# mount -r -o nosuid,bg serv:/usr/src /usr/src
```

Topic: **NETWORK CONFIGURATION**
SubTopic: **NFS**

Objective 117
Verify the appropriate entries are in the exports file.

Rationale:
NFS is a distributed database system that is designed to allow several hosts to share files over the network. One of the most common uses of NFS is to allow diskless workstations to be installed in offices, while keeping all disk storage in a central location. As distributed by Sun, NFS has no security features enabled. This means that any host on the Internet may access your files via NFS, regardless of whether you trust them or not (Curry, 1990).

Fortunately, there are several easy ways to make NFS more secure. The more commonly used methods are described in this section, and these can be used to make your files quite secure from unauthorized access via NFS. Secure NFS, introduced in SunOS Release 4.0, takes security one step further, using public-key encryption techniques to ensure authorized access (Curry, 1990).

The file /etc/exports is perhaps one of the most important parts of NFS configuration. This file lists which file systems are exported (made available for mounting) to other systems (Curry, 1990).

The root= keyword specifies the list of hosts that are allowed to have super-user access to the files in the named file system. The access= keyword specifies the list of hosts (separated by colons) that are allowed to mount the named file system. If no access= keyword is specified for a file system, any host anywhere on the network may mount that file system via NFS (Curry, 1990).

Obviously, this presents a major security problem, since anyone who can mount your file systems via NFS can then peruse them at his leisure. Thus, it is important that all file systems listed in exports have an access= keyword associated with them. Netgroups can also be specified (Curry, 1990).

Normally, NFS translates the super-user id to a special id called "nobody" in order to prevent a user with "root" on a remote workstation from accessing other people's files. This is good for security, but sometimes a nuisance for system administrators, since you cannot make changes to files as "root" through NFS (Curry, 1990).

The exports file also allows you to grant super-user access to certain file systems for certain hosts by using the root= keyword. Following this keyword a colon-separated list of up to ten hosts may be specified (Curry, 1990).

Granting "root" access to a host should not be done lightly. If a host has "root" access to a file system, then the super-user on that host will have complete access to the file system, just as if you had given him "root" password on the server. Untrusted hosts should never be given "root" access to NFS file systems (Curry, 1990).

Topic: **NETWORK CONFIGURATION**
SubTopic: **NFS**

Expected Results:
The entries in the /etc/exports file should be similar to the entry below:

```
/usr/stuff      -access=bear:dog,anon=-65534,ro
```

Only necessary filesystems are exported.
Only authorized hosts are given access to the exported filesystems.
All entries use fully qualified hostnames (Preferably an ip address).
Filesystems are shared using "anon=-1" to disallow accesses that are not accompanied by a user ID.

The NFS server is not self-referenced, either by name or by specification of a 'localhost' entry.
File systems to be exported are shared as read-only, except where specifically approved by the responsible security officer.
Only the minimum access necessary is given on the exported filesystem.
File systems to be exported are shared non-setuid.
The "root = " option should NOT be used.
Access should be granted by netgroup or host.

Comments:
Use of a network file system must be approved for use by the responsible security officer.
All NFS patches have been applied.
Ensure that you never export file systems unintentionally to the world.
Review periodically what you currently have exported.
Run fsir and for all your file systems and rerun it periodically.
Ensure that the RPC portmapper does not allow proxy requests.

Step: 2

Required Action:
Execute the following command and ensure that the owner and permissions of the exports file are correct:

```
#/usr/bin/ls -lg /etc/exports
```

Expected Results:
The file /etc/exports has permissions 644.

The file /etc/exports is owned by root.

Comments:
Use of a network file system must be approved for use by the responsible security officer.

All NFS patches have been applied.

Review periodically what you currently have exported.

Run fsir and for all your file systems and rerun it periodically.

Ensure that the RPC portmapper does not allow proxy requests.

Topic: **NETWORK CONFIGURATION**
SubTopic: **NFS**

Objective 120
Verify NFS port monitoring is enabled.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Enable NFS port monitoring. To do this, add the following lines to /etc/system:

```
set nfs:nfs_portmon = 1
```

Expected Results:

Comments:

Topic: **NETWORK CONFIGURATION**
SubTopic: **NIS**

Objective 125
Verify the "+" line in the /etc/passwd file for each NIS client has been correctly entered.

Rationale:
NIS is a distributed database system that lets many computers share password files, group files, host tables, and other files over the network. Although the files appear to be available on every computer, they are actually stored on only a single computer, called the NIS master server.

NIS allows many hosts to share password files, group files, and other files via the network, while the files are stored on only a single host. Unfortunately, NIS also contains a few potential security holes (Curry, 1990).

The plus sign tells UNIX programs that scan a system database file (such as /etc/passwd or /etc/group) to ask the NIS Server for the remainder of the file. Verify that the "+" line in the /etc/passwd file on each NIS client has been correctly entered. Verify that the "+" line has not been added to the /etc/passwd file on the NIS master server.

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Login to each NIS client, and browse the /etc/passwd file. The following line should be found:

```
+:*:0:0:::
```

Verify that the following line has not been accidentally added instead; if it has, anyone can log into the system by providing a "+" at the login prompt:

```
+::0:0:::
```

Password file, line nnn, no password: +

Expected Results:
The NIS server and clients should be correctly configured.

Comments:

Topic: **NETWORK CONFIGURATION**
SubTopic: **NIS**

Objective 126
Verify a local host's ypbind does not accept ypbind -H commands from remote hosts.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
From a networked host, type the following command:

```
% /usr/etc/yp/ypset -H <localhost name>
```

Expected Results:
The ypbind bug should be fixed.

Comments:
If the request is granted, then the version of ypbind should be replaced with a more recent version.

Topic: NETWORK CONFIGURATION
SubTopic: Penetration

Objective 89
Determine whether rusers is enabled.

Rationale:
The UNIX rusers command displays information about accounts currently active on a remote system. This may provide an attacker with account names or other information useful in mounting an attack (CERT Advisory CA-93:14).

DII COE SRS Requirement:
RUSERS

Test Actions:
Step: 1

Required Action:
Type the following command from a networked host:

```
%  rusers -a <hostname>
```

Expected Results:
If the error message "<hostname>: RPC: Program not registered," then rusers is disabled. If instead, a list of user names and login information was generated, then a rusers server is running on the host.

Comments:
rusers should not be enabled unless there is a legitimate business need.

Topic: **NETWORK CONFIGURATION**
SubTopic: **Trusted Hosts**

Objective 161
Check the /etc/hosts.equiv file to verify that the default setting of "trust all hosts" has been changed. If there are individual entries in this file, verify that all entries are appropriate.

Rationale:
One of the most convenient features of the UNIX networking software is the concept of "trusted" hosts. The software allows the specification of other hosts (and possibly users) who are to be considered trusted - remote logins and remote executions from these hosts will be permitted without requiring the user to enter a password. This is very convenient, because users do not have to type their password every time they use the network. Unfortunately, for the same reason, the concept of a trusted host is also extremely insecure (Curry, 1990).

The Internet worm made extensive use of the trusted host concept to spread itself throughout the network. Many sites that had already disallowed trusted hosts did fairly well against the worm compared with those sites that did allow trusted hosts (Curry, 1990).

The file /etc/hosts.equiv can be used by the system administrator to indicate trusted hosts. Each trusted host is listed in the file, one host per line. If a user attempts to login or execute a command remotely from one of the systems listed in hosts.equiv, and that user has an account on the local system with the same login name, access is permitted without requiring a password (Curry, 1990).

Provided adequate care is taken to allow only local hosts in the hosts.equiv file, a reasonable compromise between security and convenience can be achieved. Nonlocal hosts (including hosts at remote sites of the same organization should never be trusted. Also, if there are any machines at your organization that are installed in "public" areas you should not trust these hosts (Curry, 1990).

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
Execute the following command:

```
%ls -ldgb /etc/hosts.equiv; /bin/more /etc/hosts.equiv
```

Expected Results:
The following response is displayed:

```
/etc/hosts.equiv: No such file or directory
/etc/hosts.equiv: No such file or directory
```

Comments:
Check for the presence of /etc/hosts.equiv after each operating system or patch installation.

Step: 2

Required Action:

If the responsible security officer has approved the use of a /etc/hosts.equiv file for a specific purpose execute the following command:

```
%ls -ldgb /etc/hosts.equiv; /bin/more /etc/hosts.equiv
```

Topic: **NETWORK CONFIGURATION**
SubTopic: **Trusted Hosts**

Expected Results:
- The owner of /etc/hosts.equiv is root.
- The permissions of /etc/hosts.equiv are set to 600.
- The first character of /etc/hosts.equiv is not '-'.
- /etc/hosts.equiv does not contain a line with only a "+" (a plus sign).
- /etc/hosts.equiv lists only a small number of trusted hosts, and all hosts listed are within your domain or under your management.
- /etc/hosts.equiv does not include '!' or '#'.
- All hosts in /etc/hosts.equiv are specified using IP addresses to mitigate DNS spoof attacks.
- Use netgroups in /etc/hosts.equiv for easier management.

Comments:
The /etc/hosts.equiv file contains a list of trusted hosts, one per line. If a user attempts to log in remotely (using rlogin) or to remotely execute a command (using rsh) from one of the hosts listed in this file, and if that user has an account on the local system with the same login name, the system allows the user to log in without a password. The /etc/hosts.equiv file may have several entries. It should be verified that each entry is appropriate. A line of the form +@host-group makes all of the hosts in the network group hostgroup trusted; likewise, a line which has the form -@anotherhostgroup makes all of the hosts in the networkgroup anotherhostgroup specifically not trusted. The file is scanned from the beginning to the end; the scanning stops after the first match. A single line of + in the hosts.equiv file indicates that every known host is trusted. This can create a serious security problem. It is recommended that the /etc/hosts.equiv file be removed

Topic: **NETWORK CONFIGURATION**
SubTopic: **promiscuous ethernet interface**

Objective 280
Verify that no interface is in promiscuous mode.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
An Ethernet interface that is running in promiscuous mode can be identified with the following command:

```
/usr/etc/ifconfig -a | grep -i promisc
```

Expected Results:
Any output is an indication of an ethernet interface in promiscuous mode. This is usually a bad sign and the system should be examined closely to determine if ethernet sniffers are being run on the system.

Comments:
An interface in promiscuous mode will allow programs to read passwords and other data that should be kept secret from the network.

Topic: **SECURE TERMINALS**
SubTopic:

Objective 92
Determine if the "secure" terminal feature is used on the system.

Rationale:
Under newer versions of UNIX, the concept of a "secure" terminal has been introduced. Simply put, the super-user (root) may not log in on a nonsecure terminal, even with a password. (Authorized users may still use the su command to become super-user, however.) The file /etc/ttytab is used to control which terminals are considered secure. The keyword "secure" at the end of each line in /etc/ttytab indicates that the terminal is considered secure (Curry, 1990).

The most secure configuration is to remove the "secure" designation from all terminals, including the console device. This requires that those users with super-user authority first log in as themselves, and then become the super-user via the su command. It also requires the "root" password to be entered when rebooting in single-user mode, in order to prevent users from rebooting their desktop workstations and obtaining super-user access. This is how all diskless client machines should be set up (Curry, 1990).

Secure terminals can be declared in /etc/ttys by appending the word "secure" to the terminal's definition.

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
Browse /etc/ttys. Determine if the word "secure" appears at the end of the line for any of the terminal definitions.

Expected Results:
At most, only the operator's console should be defined as "secure."

Comments:

Topic: SYSTEM ARCHITECTURE
SubTopic:

Objective 143
Verify development tools such as language compilers, linkers, and debuggers are adequately protected and can only be accessed by authorized users.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
For each development tool, enter:

```
ls -alg <development tool>
```

where <development tool> takes the following values:

/usr/bin/adb
/usr/bin/as
/usr/bin/bc
/usr/lib/compile
/usr/bin/cb
/usr/bin/cflow
/usr/bin/cxref
/usr/bin/dbxtool
/usr/bin/ld
/usr/bin/lex
/usr/bin/m4
/usr/bin/od
/usr/bin/rpcgen
/usr/bin/yacc
/usr/ucb/dbx
/usr/ucb/gcore
/usr/ucb/sccs
/usr/ucb/xstr
/usr/5lib/compile
/usr/5bin/lint
/usr/5bin/od

Expected Results:

Comments:

Topic: **SYSTEM ARCHITECTURE**
SubTopic:

Objective 158

Verify the user environment is configured properly. By default, the /etc/profile file sets the user terminal type, checks for new email, and sets the umask. Any other activity should be explicitly approved.

Rationale:

DII COE SRS Requirement:

Test Actions:

Step: 1

Required Action:

As root execute the following shell script for printing the umask value for each user:

```
#!/bin/sh
date
uname -a
PATH=/bin:/usr/bin:/usr/etc:/usr/ucb

HOMEDIRS=`cat /etc/passwd | awk -F: 'length($6)>0 {print $6}' | sort -
u`
FILES=".cshrc .login .profile "
for dir in $HOMEDIRS
do
         echo "----------------------------------------"
         echo Home Directory being checked is $dir
      for file in $FILES
      do
         ls -ald $dir/$file
         if [ -f $dir/$file ]
         then
            grep -s umask /dev/null $dir/$file
         fi
      done
done
echo "----------------------------------------"
```

Expected Results:

The umask value for each user is set to something sensible like 027 or 077.

Comments:

SCRIPT DOES NOT WORK UNDER NIS or NIS+

When a file or directory is created, it has a default set of permissions. These default permissions are determined by the value of umask in the system file /etc/profile, or in a user's .cshrc or .login file. By default, the system sets the permissions on a text file to 666, granting read and write permission to user, group, and others, and to 777 on a directory or executable. The value assigned by umask is subtracted

from the default. This has the effect of denying permissions in the same way that chmod grants them. If possible, a .cshrc, .login, and .profile should be created for each user owned by root and readable by the user with correct environment settings.

Topic: **SYSTEM ARCHITECTURE**
SubTopic:

***Step:* 2**

Required Action:
Utilize the following shell script for viewing the account initialization files for each user:

```
#!/bin/sh
date
uname -a
PATH=/bin:/usr/bin:/usr/etc:/usr/ucb

HOMEDIRS=`cat /etc/passwd | awk -F: 'length($6)>0 {print $6}' | sort -
u`
FILES=".cshrc .login .profile .logout .mwmrc .Xsession .Xdefaults .exrc
.forward .rhosts"
for dir in $HOMEDIRS
do
         echo "----------------------------------------"
         echo Home Directory being checked is $dir
      for file in $FILES
      do
         ls -ald $dir/$file
         if [ -f $dir/$file ]
         then
            more $dir/$file
         fi
      done
done
echo "----------------------------------------"
```

Expected Results:
All account initialization files in user $HOME, and the default files that are used if these files are not present, have been reviewed to ensure that only acceptable actions are taken. Acceptable actions include: set user terminal type, check for new e-mail, and set a proper umask (027 or 077). Any other actions should be explicitly approved by the responsible security officer. All user account initialization files are owned by the user (or root) and have permissions 640.

Comments:
SCRIPT DOES NOT WORK UNDER NIS or NIS+. [Acceptable actions for .mwmrc and .Xsession TBD.]
If possible, a .cshrc, .login, and .profile should be created for each user owned by root and readable by the user with correct environment settings.

Topic: **SYSTEM ARCHITECTURE**
SubTopic:

Step: ***3***

Required Action:
Ensure the default account initialization files are secure. A shell script for viewing the files follows:

```
#!/bin/sh
date
#!/bin/sh
date
uname -a
echo ----------------------------------------------
echo /etc/.profile
echo ----------------------------------------------
ls -al /etc/.profile
cat /etc/.profile

echo ----------------------------------------------
echo /etc/skel/local.cshrc
echo ----------------------------------------------
ls -al /etc/skel/local.cshrc
cat /etc/skel/local.cshrc
echo ----------------------------------------------
echo /etc/skel/local.login
echo ----------------------------------------------
ls -al /etc/skel/local.login
cat /etc/skel/local.login

echo ----------------------------------------------
echo /etc/skel/local.profile
echo ----------------------------------------------
ls -al /etc/skel/local.profile
cat /etc/skel/local.profile

echo ----------------------------------------------
echo /etc/profile
echo ----------------------------------------------
ls -al /etc/profile
cat /etc/profile

echo ----------------------------------------------
echo ----------------------------------------------
echo DII COE initialization files
echo ----------------------------------------------
echo ----------------------------------------------
echo /etc/csh.login
echo ----------------------------------------------
ls -al /etc/csh.login
cat /etc/csh.login
```

Topic: **SYSTEM ARCHITECTURE**
SubTopic:

```
echo ----------------------------------------------
echo /etc/dt/config/sys.dtprofile
echo ----------------------------------------------
ls -al /etc/dt/config/sys.dtprofile
cat /etc/dt/config/sys.dtprofile

echo ----------------------------------------------
echo /h/USERS/local/sysadmin/Scripts/.cshrc
echo ----------------------------------------------
ls -al /h/USERS/local/sysadmin/Scripts/.cshrc
cat  /h/USERS/local/sysadmin/Scripts/.cshrc

echo ----------------------------------------------
echo /h/USERS/local/sysadmin/Scripts/.login
echo ----------------------------------------------
ls -al /h/USERS/local/sysadmin/Scripts/.login
cat  /h/USERS/local/sysadmin/Scripts/.login
 echo ----------------------
echo /h/COE/Scripts/.cshrc.COE
echo ------
ls -al /h/COE/Scripts/.cshrc.COE
cat /h/COE/Scripts/.cshrc.COE
echo ----------------------------------------------
echo /h/COE/Scripts/.login.COE
echo ----------------------------------------------
ls -al /h/COE/Scripts/.login.COE
cat /h/COE/Scripts/.login.COE

echo ----------------------------------------------
echo /h/COE/Scripts/.xsession.COE
echo ----------------------------------------------
ls -al /h/COE/Scripts/.xsession.COE
cat /h/COE/Scripts/.xsession.COE

echo ----------------------------------------------
echo    $COE_HOME/Scripts
echo ----------------------------------------------
ls -alg  $COE_HOME/Scripts
echo ----------------------------------------------
```

Expected Results:
All default account initialization files that are used if user account initialization files are not present have been reviewed to ensure that only acceptable actions are taken. Acceptable actions include: set user terminal type, reviewed to ensure that only acceptable actions are taken. Acceptable actions include: set user terminal type, check for new e-mail, and set a proper umask (027 or 077). Any other actions should be explicitly approved by the responsible security officer.

The default account initialization files are owned by root and have permissions 644.

Topic: **SYSTEM ARCHITECTURE**
SubTopic: **Operating System**

Comments:
/etc/profile allows the system administrator to perform services for the entire user community. The file $HOME/.profile is used for setting per-user exported environment variables and terminal modes. Care must be taken in providing system-wide services in /etc/profile.

Step: 4

Required Action:
As root, execute the following command:

```
/bin/find / -name ".exrc"  -print -exec ls -ld {} \; \
-exec /usr/bin/more {} \;
```

Expected Results:
There are no .exrc files on the system or the "exrc" option for each user is set to "noexrc".

Comments:
The editing environment defaults to certain configuration options. When an editing session is initiated, vi attempts to read the EXINIT environment variable. If it exists, the editor uses the values defined in EXINIT, otherwise the values set in $HOME/.exrc are used. If $HOME/.exrc does not exist, the default values are used.

To use a copy of .exrc located in the current directory other than $HOME, set the exrc option in EXINIT or $HOME/.exrc. Options set in EXINIT can be turned off in a local .exrc only if exrc is set in EXINIT or $HOME/.exrc.

Topic: **SYSTEM ARCHITECTURE**
SubTopic: **Operating System**

Objective 151
Verify the system kernel configuration file is correct.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
Review the file:

```
/etc/system.
```

Expected Results:

Comments:

Topic: **SYSTEM ARCHITECTURE**
SubTopic: **Operating System**

Objective 153
Verify the appropriate operating system patches have been applied.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Verify that the appropriate operating system patches have been applied.

```
#  showrev -a
```

Current operating system patch recommendations can be obtained from the following site:

http://support.mayfield.hp.com

Expected Results:

Comments:

Topic: **SYSTEM ARCHITECTURE**
SubTopic: **Printer Definition**

Objective 154
Verify only appropriate printers are defined.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
As root execute the following command:

```
#/usr/bin/sam
```

Expected Results:

Comments:

Step: 2
Required Action:
Select "Printers and Plotters."

Expected Results:
The list of printers should contain only authorized printers.

Comments:

Topic: **System Architecture**
SubTopic: **Telnet bug**

Objective 278
Verify that the telnet bug does not exist.

Rationale:
There is a security hole in some versions of telnet that will allow any user on the system to overwrite any file. Using the command will overwrite any file in any filesystem with a zero-length root-owned file.

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
As root, execute the following commands:

```
#cd /tmp
#/usr/ucb/vi file1

    insert some text
```

Save the file, exit the editor, and type the following commands:

```
#ls /tmp/file1
#/usr/ucb/more /tmp/file1
```

Expected Results:
The file size of /tmp/file1 is larger than 0 and the text inserted into file1 is displayed on the screen.

Comments:

Step: 2

Required Action:
As an unprivileged user, execute the following command:

```
$/usr/bin/telnet -n /tmp/file1 localhost
$ls /tmp/file1
```

Expected Results:
The file size is NOT 0.

Comments:
If the file size of /tmp/file1 is 0, the telnet daemon must be replaced.

Topic: **SYSTEM ARCHITECTURE**
SubTopic: **Operating System**

Objective 152
Determine the OS version installed. Verify that it is the correct version.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Type in the following command:

```
#uname -a
```

Expected Results:
Output similar to the following is printed to the screen:

```
HP-UX oban A.10.0 E 9000/712 2011035351 8-user license
```

Comments:
The most important parts are the "HP-UX" and the "10.0" portions that indicate that the host being tested is running the HP-UX version 10 operating system.

Topic: **TELNET**
SubTopic:

Objective 160
Verify a user is always prompted for a password when telneting into the host machine.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Logon to a test account. Attempt to telnet by typing the command `"telnet localhost"`. The system should respond with the login prompt. Enter a valid username.

Expected Results:
Should be prompted for a password.

Comments:

Topic: **TRUSTED HOSTS**
SubTopic:

Objective 162
Verify the entries in the hosts.lpd file are appropriate.

Rationale:
One of the most convenient features of the UNIX networking software is the concept of "trusted" hosts. The software allows the specification of other hosts (and possibly users) who are to be considered trusted - remote logins and remote executions from these hosts will be permitted without requiring the user to enter a password. This is very convenient, because users do not have to type their password every time they use the network. Unfortunately, for the same reason, the concept of a trusted host is also extremely insecure (Curry, 1990).

The Internet worm made extensive use of the trusted host concept to spread itself throughout the network. Many sites that had already disallowed trusted hosts did fairly well against the worm compared with those sites that did allow trusted hosts (Curry, 1990).

Normally, the UNIX lpd system allows only trusted hosts to print on a local printer. However, if a host name is placed in the /etc/hosts.lpd file, users on that system are allowed to print to the local printer.

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
As root, execute the following commands:

```
#more /etc/hosts.lpd
#more /usr/spool/lp/.rhosts
```

Expected Results:
All entries in the lpd file are appropriate (i.e. only hosts that should access the local printers are listed).

Comments:

Topic: **UUCP**
SubTopic: **Penetrate**

Objective 172
Verify known UUCP bugs have been fixed.

Rationale:
UUCP is one of the oldest major subsystems of UNIX, and has had its share of security holes. All of the known security problems have been fixed in recent years. Unfortunately, there are still many old versions of UUCP in use.

DII COE SRS Requirement:
Test Actions:
Step: 1

Required Action:
Perform the following tests of UUCP:

The mail system should not allow mail to be sent directly to a file. Test whether the system allows mail to be
sent to a file with the command sequence:

```
$ mail /tmp/mailbug
 this is a mailbug file test
 ^D
```

Expected Results:
If the file mailbug appears in the /tmp directory, then the mailer is unsecure.

Comments:
Remove and replace the uucp software.

Step: 2

Required Action:
As a non privileged user, execute the following command sequences:

```
$ uux - mail 'root `/bin/touch /tmp/foo`'
this is a mailbug command test
^D
$ uux - mail 'root & /bin/touch /tmp/foo'
this is another test
^D
```

Expected Results:
Mail should be returned saying that `/bin/touch /tmp/foo` is an unknown user. If the mailer executed the touch (a foo file will be created in the /tmp directory), the uux program is unsecure.

Comments:
The UUCP system should not allow a command to be encapsulated in addresses to prevent system execution of commands encapsulated in addresses.

Topic: **UUCP**
SubTopic: **Penetrate**

Step:* *3

Required Action:
As a non privileged user, execute the following command sequences:

```
   $ uux - mail ‘root & /bin/touch /tmp/foo`’
 this is another mailbug command test
   ^D

   $ uux - mail ‘root & /bin/touch /tmp/foo’
   this is another test
   ^D
```

Mail should be returned saying that `/bin/touch /tmp/foo` is an unknown user. If the mailer executed the touch:(a foo file will be created in the /tmp directory) then the uux program is unsecure.

Comments:
The UUCP system should not allow a command to be encapsulated in addresses to prevent system execution of commands encapsulated in addresses.

Topic: **UUCP**
SubTopic: **Disabled**

Objective 288
Verify that uucp is not enabled.

Rationale:
UUCP is one of the oldest major subsystems of UNIX, and has had its share of security holes. All of the known security problems have been fixed in recent years. Unfortunately, there are still many old versions of UUCP in

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Review the following files:

```
/etc/inetd.conf
```

Expected Results:
The uucp entry in /etc/inetd.conf should NOT be enabled (i.e., the first character on the line for uucp should be a "#").

Comments:
UUCP is one of the oldest major subsystems of UNIX, and has had its share of security holes. Although the design is not secure, the known security holes have been fixed in recent years. Unfortunately, there are still many old versions of UUCP in use.

Step: 2
Required Action:
Verify that the files are owned by uucp using the following commands:

```
%  ls -lgdb /etc/uucheck
%  ls -lgdb /etc/uucico
%  ls -lgdb /etc/uucp
%  ls -lgdb /etc/uux
```

Expected Results:
The files are owned by uucp.

Comments:
The uucp programs run SUID uucp, not SUID root. Other than being able to read the spooled UUCP files, the uucp user doesn't have any special privileges.

Topic: **UUCP**
SubTopic: **Disabled**

Step: 3

Required Action:
Verify USERFILE is owned by uucp and not world writeable. Verify that it contains the correct entries. Verify that the file contains appropriate entries. Type the following command:

```
% ls -lgdb /usr/lib/uucp/USERFILE
```

Verify that the file is not world writeable.

The format for USERFILE varies by vendor. Reference the HP documentation to determine the correct formatting for this system. Verify that /usr/lib/uucp (or /et/uucp) is not in the permission list.

Expected Results:
/usr/lib/uucp/USERFILE is owned by root and is not world writeable./usr/lib/uucp should not be in the USERFILE permission list.

Comments:
The /usr/lib/uucp/USERFILE file controls which files on your computer can be accessed through the UUCP:system. Normally, you specify one entry in USERFILE for each UUCP login in the /etc/passwd file.

Step: 4
Required Action:
Verify L.sys is owned by uucp and is not world readable or world writeable by executing the following command:

```
% ls -lgdb /usr/lib/uucp/L.sys
```

Expected Results:
L.sys is owned by uucp and is NOT world readable or writeable.

Comments:
Because it logs in to remote systems, uucico has to keep track of the names, telephone numbers, account names, and passwords it uses to log into these machines. This information is kept in a special file called /usr/lib/uucp/L.sys.

Step: 5

Required Action:
Verify L.cmds is owned by root (not by uucp) and is world readable by executing the following command:

```
% ls -lgdb /usr/lib/uucp/L.cmds
```

Expected Results:
L.cmds is owned by root and is world readable.

Topic: UUCP
SubTopic: Disabled

Comments:
UUCP is the Unit-to-Unix CoPy system, a collection of programs that provide rudimentary networking for UNIX computers. UUCP allows files to be copied from computer to computer and allows commands to be remotely executed.

Step: 6

Required Action:
Determine if the system has enabled UUCP callback.

Expected Results:
UUCP callback is enabled if possible.

Comments:
Version 2 UUCP has a callback feature that can be used to enhance security. With callback, when a remote system calls the local computer, the system immediately hangs up on the remote system and calls back. No special callback hardware is required to take advantage of UUCP callback, because it is performed by the system software, not by the modem. Note that only one system out of each pair of communicating systems can have callback enabled.

Step: 7

Required Action:
Verify uucp's home directory is in an appropriate directory using the following commands:

```
$grep uucp /etc/passwd
$ls -lgd  `grep uucp /etc/passwd | awk -F: 'length($6)>0 {print $6}'`
```

Expected Results:
The uucp home directory should not be in a directory that is world writeable.

Comments:
The home directory for the uucp account should not be in the directory /usr/spool/uucp/uucppublic, or any other directory that can be written to by a uucp user.

Step: 8

Required Action:
Use the following command to ensure that there is no .rhosts file in the uucp home directory:

```
#find  `grep uucp /etc/passwd | awk -F: 'length($6)>0 \
{print $6}'`  -name .rhosts -exec ls -ldb {} \;
```

Ensure that no uucp owned files or directories are world writeable.

Expected Results:
There should be no output from this command.

Comments:

Topic: UUCP
SubTopic: Disabled

Step: 9

Required Action:
As root ensure that no uucp owned files or directories are world writeable using the following command:

```
find / -user uucp -perm -2 -exec ls -ldb {} \;
```

Expected Results:
There is no output indicating no files on the system that are owned by uucp and world writeable.

Comments:

Topic: **WWW-HTTPD**
SubTopic:

Objective 175
Verify the http server daemon is not being run as root, but as a specially created nonprivileged user such as “httpd”.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
As root, execute the following command:

```
#/bin/find / -name "*http*" -exec ls -ldb {} \;
```

Expected Results:
File permission listing reveals that the owner of the http server daemon (usually httpd) is not root and not SUID, but as a specially created nonprivileged user such as "httpd."

Comments:

Topic: **WWW-HTTPD**
SubTopic:

Objective 176
Verify httpd client processes are not being run as root.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Use the following command to verify that the http client applications are not being run as root:

```
#/bin/find / -name "*osaic*" -exec ls -ldb {} \;
#/bin/find / -name "*etscape*" -exec ls -ldb {} \;
```

Expected Results:
The file permissions on all http clients listed are not owned by root and are not SUID.

Comments:

Topic: **X WINDOW SYSTEM**
SubTopic: **Use of xauth access control**

Objective 183
Verify the system uses the xauth X server access control mechanism instead of the xhosts mechanism.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
As root, execute the following command:

```
/bin/find /etc -name "*rc*" -type f  -exec ls -lgdb {} \;
-exec /bin/grep xdm  {} \;
```

Expected Results:
xdm is initiated with -auth $HOME/.Xauthority.

Comments:

Step: 2
Required Action:
As an unprivileged user, execute the following command:

```
echo $XAUTHORITY
```

Expected Results:
This variable should exist and contains the magic cookie used to authenticate valid users attempting to connect to the X server. If xauth is being used and this variable is not present, then the $HOME/.Xauthority file contains the magic cookie (this is not as secure).

Comments:

Step: 3
Required Action:
As root, execute the following command:

```
/bin/find / -name xdm-config -exec ls -lgdb {} \;
 -exec /usr/ucb/more {} \;
```

Expected Results:
The following lines are included:

```
DisplayManager*authorize:    true
DisplayManager*authname:   XDM-AUTHORIZATION-1
```

Comments:

The first line turns on authorization for all X servers controlled by a given xdm program.
The second line sets the authority scheme to XDM-AUTHORIZATION-1.

Topic: **X WINDOW SYSTEM**
SubTopic:

Objective 68
Ensure the setuid and setgid privilege bits are not set on the xterm program.

Rationale:
X is a popular network-based window system that allows many programs to share a single graphical display. The X Window System is a major security hazard. Although there are a number of mechanisms inside X to give some security features, these can be circumvented in many circumstances (Garfinkel and Spafford, 1992).

DII COE SRS Requirement:

Test Actions:
Step: 1

Required Action:
As root, execute the following command:

```
/bin/find / -name xterm -exec ls -ldg {} \;
```

Expected Results:
The xterm program is not SUID or SGID.

Comments:
On DII COE perform the same command substituting dtterm for xterm.

Topic: **X WINDOW SYSTEM**
SubTopic:

Objective 178
Verify xterm logging is not being performed.

Rationale:

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Click and hold the right mouse button down and start an xterm session. In the xterm window execute the following command:

```
ps -ef | grep xterm
```

Expected Results:
NO xterm session entry contains the -l switch.

Comments:

Topic: **X WINDOW SYSTEM**
SubTopic:

Objective 179
Verify the systems listed in xhost are appropriate. Determine what release of X is used on the system.

Rationale:
X uses a system called xhost to provide a minimal amount of security for window system users. Each X Window Server has a built-in list of hosts from which it will accept connections; connections from all other hosts are refused. The design of the X Window System allows any client that successfully connects to the X Window Server to exercise complete control over the display. If a person can log into a system, they can capture another user's keystrokes no matter how the xhosts is set (Garfinkel and Spafford, 1992).

Release 4 of the X Window Protocol has a secure feature on the xterm command that makes the window change its color if it is not receiving its input directly from the keyboard. This is a partial fix, but it is not complete (Garfinkel and Spafford, 1992).

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
Type the following command to produce a list of which hosts are listed in xhost:

```
%  xhost
```

Expected Results:
Only trusted hosts should be in list returned or the message "Access control enabled, only authorized clients can connect." will be returned.

Comments:
It is preferable that the xhost security not be used and that xauth or another security mechanism be used.

Topic: **X WINDOW SYSTEM**
SubTopic: **Denial of Service**

Objective 182
Determine if the X server is vulnerable to the specified denial of service attack.

Rationale:
Even if the xhost facility is used, the X Window System may be vulnerable to attack from computers not in the xhost list. The X11R3 Window Server reads a small packet from the client before it determines whether or not the client is in the xhost list. If a client connects to the X Server but does not transmit this initial packet, the X Server halts all operation until it times out in 30 seconds (Garfinkel and Spafford, 1992).

DII COE SRS Requirement:

Test Actions:
Step: 1
Required Action:
From a networked host, type the following command:

```
%  telnet <localhost> 6000
%  telnet <localhost> 6001
```

Expected Results:
Should get a message "Unable to connect".

If the X server has a problem, the workstation's display will freeze. In some X implementations, the X server will time out after 30 seconds and resume normal operations. Under other X implementations, the server will remain blocked until the connection is aborted.

Comments:
The denial of service vulnerability should not exist.